Distillation-guided Image Inpainting

Maitreya Suin[1] Kuldeep Purohit[2] A. N. Rajagopalan[1]
[1] Indian Institute of Technology Madras, India [2] Michigan State University, USA
maitreyasuin21@gmail.com, kuldeeppurohit3@gmail.com, raju@ee.iitm.ac.in

Abstract

Image inpainting methods have shown significant improvements by using deep neural networks recently. However, many of these techniques often create distorted structures or blurry inconsistent textures. The problem is rooted in the encoder layers' ineffectiveness in building a complete and faithful embedding of the missing regions from scratch. Existing solutions like course-to-fine, progressive refinement, structural guidance, etc. suffer from huge computational overheads owing to multiple generator networks, limited ability of handcrafted features, and sub-optimal utilization of the information present in the ground truth. We propose a distillation-based approach for inpainting, where we provide direct feature level supervision while training. We deploy cross and self-distillation techniques and design a dedicated completion-block in encoder to produce more accurate encoding of the holes. Next, we demonstrate how an inpainting network's attention module can improve by leveraging a distillation-based attention transfer technique and further enhance coherence by using a pixel-adaptive global-local feature fusion. We conduct extensive evaluations on multiple datasets to validate our method. Along with achieving significant improvements over previous SOTA methods, the proposed approach's effectiveness is also demonstrated through its ability to improve existing inpainting works.

1. Introduction

Image inpainting is aimed at filling damaged or substituting undesired areas of images with plausible and fine-grained contents. It has a broad range of applications in fields of restoring damaged photographs, retouching pictures, etc. Early conventional works typically use low-level features hand-crafted from the incomplete input image and resort to priors (e.g., image statistics) or auxiliary data (e.g., external image databases). They either propagate low-level features from surroundings to the missing regions following a diffusive process [30, 2] or fill holes by searching and fusing similar patches from the same image or external image databases [27]. Although these methods have good effects in the completion of repeating structures, they are restricted by the available image statistics and cannot produce novel image contents. In recent years, deep learning based methods have been reported to surmount these limitations by utilizing large volumes of training images. However, many of these methods suffer from the severe ill-posedness of the task. The hole regions are entirely empty, and without sufficient guidance, neural networks struggle to reconstruct the missing contents from scratch satisfactorily.

Works like [26, 14, 17] deploy a single generative model to solve the task. To handle the inherent ill-posedness, a large majority of works uses some form of guidance at inference time. We can broadly divide these methods into two categories: (a) Coarse-to-fine: One group of works [40, 41, 20] deploys a two-stage architecture to do content formation and texture refinement separately in a step-by-step manner. These methods typically produce an intermediate coarse image with recovered structures in the first stage and send it to the second stage for texture generation. Another group of works tries to inpaint the missing region in a progressive manner using a single network [11, 16, 44]. (b) Structural guidance: Recently, [24] used edge generator within the two-stage architecture. [37] proposed a contour generator instead of edge. Such methods suffer from several limitations: (a) It takes many more parameters to deploy two generators. Methods like [37] require even more parameters for the structure prediction branch. Progressive or recurrent approaches typically suffer from slow inference speed or high computational cost. (b) Although structural knowledge improves performance, it is still limited due to the handcrafted choice of the auxiliary information. For example, the optimal structure information might vary from one scene to another (edge vs. contour).

The inpainted image quality depends heavily on the coarse network, and [32] experimentally verified that if we remove the coarse network and train end-to-end only the final network of such two-stage methods, it results in a notable drop in performance. In this work, we argue that the root cause lies in how existing networks are trained. We empirically show that when training a single network, using

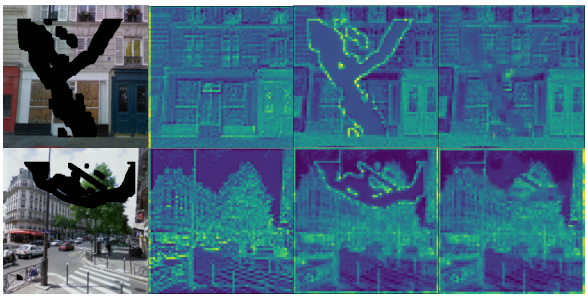

Figure 1: Visualization of encoder features with and without distillation.

the ground-truth image as the only supervision at the end fails to act as a strong enough regularizer, especially for the initial layers (Fig. 1, 3rd column). Such weaknesses propagate throughout the network resulting in poor inpainting quality. Instead of using multiple networks or guidances at inference, we show that if we can guide a network accurately while training, even a single efficient network can learn to solve the task much better without requiring a high testing-load. We propose a distillation-guided training strategy where we guide different layers of our network individually to converge to a much better optima and show that it has the potential to improve existing works as well.

We use the widely used encoder-decoder architecture [26, 17] as our backbone. The encoder is typically expected to generate a meaningful encoding for the hole regions, while in the decoder, we refine that information using dedicated tools like attention modules [40, 45]. We deploy two networks: an auxiliary network (AN) and an inpainting network (IN), where both have a similar encoder-decoder backbone with three levels. The AN is used only for training to provide accurate information on what the missing regions should contain. We start with an under-complete autoencoder as our AN, which takes the ground truth image as input and tries to produce the same as output. The intuition is that its features will be uncorrupted and can be used to supervise the inpainting encoder. As the training progresses, we further finetune the AN using meta-networks to produce more suitable uncorrupted features to help with the inpainting task. We conceptually divide AN and IN at every down/up-sampling level and each of these sub-networks (representing all the encoder/decoder layers at that particular level) of the IN receives a supervisory signal from the same sub-network of the AN as guidance. With these supervisions, we use knowledge distillation to make the IN learn from the "ideal" features of the AN.

Knowledge distillation (KD), introduced initially in the deep learning setting [13], is a technique that transfers knowledge from one architecture (teacher) to another (student). We use the AN as the 'teacher' network that provides supervisory signals for different layers of the 'student' IN. For encoder, we use feature maps of AN to inform the IN about the ideal embeddings of the holes. To make sure that the IN encoder is able to mimic this 'ideal' target as close as possible, we propose a dedicated 'completion block.' It solely focuses on filling the holes by adaptively gathering relevant information from the neighborhood. We show that completion block coupled with AN's intermediate supervision brings significant performance improvement to an encoder's performance (Fig. 1, 4th column).

After generating a coarse embedding of the holes, we focus on refining it in the decoder. Attention modules are de facto standard for feature refinement, which inherently assumes that the missing regions in the input feature are roughly complete, which is needed to generate a valid pixel/patch-wise similarity score. This assumption can fail for complex holes, hole boundaries, etc. We design a distillation-based attention transfer technique, where we force the attention module of IN to learn the ideal pixel/patch-wise similarities from the same module in the AN. Further, to generate more refined results, we introduce a pixel-adaptive global-local feature fusion technique, which allows each hole pixel to generate content consistent with the immediate local neighborhood and boundaries.

Our main contributions are as follows:

(1) We propose a distillation based training strategy for inpainting. For encoder, we demonstrate the utility of deep feature level supervision coupled with dedicated adaptive completion-blocks that generates much better embedding of the hole regions. (2) For decoder, we design an attention transfer technique that enables the attention module to learn an ideal affinity-finding behavior for refining the coarse embeddings from the encoder. Further, we design a pixel-adaptive global-local consistent structure for generating more coherent results. (3) The proposed training strategy directly helps the inpainting network to learn better and does not require multiple generators or progressive refinement at inference time, increasing efficiency. Our network achieves SOTA score on 3 standard datasets. We also verify the efficacy of the proposed distillation strategy by demonstrating its role in improving existing SOTA methods.

2. Related Works

Image Inpainting: Deep learning-based image inpainting approaches [17, 26] are generally based on generative adversarial networks (GANs) to generate the pixels of a missing region. [38] and [40] devised feature shift and contextual attention operations, respectively, to allow the model to borrow feature patches from distant areas of the image. [20] used a coherent semantic attention layer to ensure semantic relevance between swapped features. [37, 24] filled images with contour/edge completion and image completion in a step-wise manner. [29] first predicted smooth structure and used that for the final stage. [44] used cas-

caded generators to progressively fill in the image. These approaches attempted to solve inpainting tasks by adding structural constraints, but they still suffer from the limitation of handcrafted guidance, huge load of two generators and lack of local semantic consistency.

Distillation: Distillation technique is mainly used in image classification [13], image segmentation [21], multiple object detection [4], speech recognition [38] and reading comprehension [14], etc. A large quantity of approaches have been proposed to reinforce the efficiency of student models' learning capability as the student learns from more informative sources. There are several other variants of this technique like [29] that extend this idea to train a student model which not only learns from the outputs of the teacher but also uses the intermediate representations learned by the teacher as additional guidance. [34] proposed ensemble of teachers, [23] showed cascaded distillation. Few recent works [1, 9] have shown that distilling from an identical teacher network trained on the exact same task (self-distillation) can also significantly improve the student. In this work, addressing the diverse needs of different parts of the network, we deploy three variants of distillation: (a) Distillation between two networks' encoder-features (b) Distillation between same network's encoder-features (c) Distillation between attention module's affinity-finding behavior (instead of directly distilling the decoder features).

3. Method

Our overall architecture is shown in Fig. 2. We use a standard encoder-decoder architecture as the backbone of both IN and AN, following its wide use in existing works [26, 17]. Every level of the IN encoder contains stack of standard convolution layers followed by an adaptive filtering block (Completion Block, Sec 3.1). Similarly, every level of the decoder in IN consists of convolutional layers followed by an attention module and adaptive filtering block. These two modules in the decoder are used in parallel and the outputs are dynamically fused for each pixel (Sec. 3.3). The AN has a similar architecture, except that it does not contain any adaptive filtering block, which is mainly used in the IN for filling holes. The input to the IN and AN branch are the masked and ground-truth images, respectively. We have provided the layerwise details in the supplementary material. We first discuss deep feature level supervision in the encoder, followed by the guiding mechanism in the decoder.

3.1. Intermediate Supervision in Encoder

The supervision in the encoder can intuitively be divided into two parts: (a) What to Fill. (b) How to fill. First, we discuss the distillation-based loss function added to every level of the IN-encoder. We design two types of distillation losses - (a) Cross-Distillation (CD), which are added between the same encoder levels of IN and AN, (b) Self-Distillation (SD): which are added between different levels (deep to shallow) of the IN itself. We describe CD and SD in detail next (i.e., What to Fill), followed by the description of a dedicated completion module (i.e., How to fill).

Cross-Distillation (CD): Information transferability in different layers was explored in [39]. [10] argued that if the student could produce comparable features as teacher, it should be able to perform similarly. For CD, the student IN-encoder tries to mimic the behavior of the teacher AN-encoder. If the outputs from the l^{th} layer of the AN encoder and inpainting encoder are x_l^* and x_l, then a regularizing term can be defined as:

$$R(\theta, x)^{l,l} = ||(x_l - \gamma(x_l^*)) \odot M||_2^2 \tag{1}$$

where x is the input, and θ represents the parameters of the inpainting network, M is the mask with value '1' for the missing regions and '0' elsewhere. γ represents a meta network [15]. Instead of directly mimicking uncorrupted feature x_l^* from the AN, we allow our network to learn a transformation γ representing convolutional operation, to make it more helpful for the inpainting task. Similar motivation can be extracted from inductive transfer learning settings [25], where the target domain is identical to the source domain, and the target task is different from the source task. The teacher-student pair has to be chosen such that the inpainting network is not over-regularized and stuck in a non-optimal minima. Using a similar encoder-decoder structure for both the modules, we use the down-sampling layers of the network as the breakpoints. However, considering the difference in tasks, every feature channel of the AN encoder might not be equally beneficial. Thus, we use another set of meta-networks (ρ) consisting of fully-connected layers, to decide which feature channels of the AN model are useful and relevant for the inpainting task. If there are L encoder levels, then for each of the L pairs, we introduce transfer importance predictor, which enforces different penalties for each channel according to their utility for the target task. For each pair, the cross distillation loss can be written as

$$R(\theta, x, \rho^l)^l_{cross} = \sum_{c \in C} \rho_c^l(x_l^*) ||(x_l - \gamma(x_l^*))_c \odot M||_2^2 \tag{2}$$

where $l \in \{1, L\}$ and $\rho_c^l : \mathbb{R}^{C \times H \times W} \rightarrow \mathbb{R}^C$ is the non-negative weight of channel c with $\sum_{c \in C} \rho_c^l = 1$.

Self-Distillation (SD): For self-distillation ([43, 12]), the deeper layers of the IN act as the teacher, and the shallower layer of IN act as student, which, in turn, forces the shallower layers to generate better content for the hole regions. Intuitively, such a strategy directly resonates with inpainting task as the deeper layers of the same inpainting network contain more complete information for the hole regions. Note that the features in different depths have different sizes, so we add extra layers to align them while train-

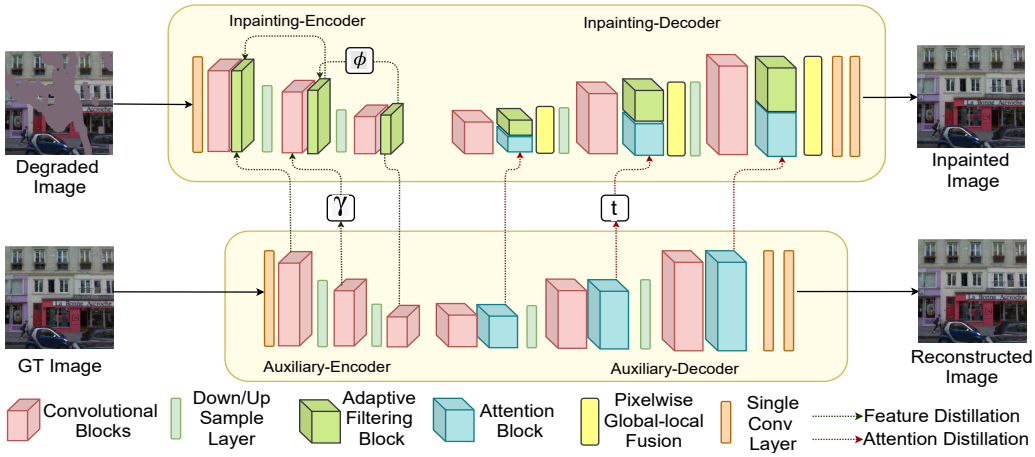

Figure 2: An overview of our method. Top branch shows the inpainting network and bottom branch shows the auxiliary network. For simplicity, each meta network is shown only for a single layer.

ing. If x_l is the encoder feature from the l^{th} level, the self distillation loss can be expressed as

$$R(\theta, x, \phi^l)^l_{self} = \sum_{c \in C} \phi^l_c(x_d)||(f_l(x_l) - x_{l+1})_c \odot M||^2_2 \quad (3)$$

where $l \in \{1, L-1\}$ and $\phi^l_c : \mathbb{R}^{C \times H \times W} \rightarrow \mathbb{R}^C$ is the non-negative weight of channel c with $\sum_{c \in C} \phi^l_c = 1$, f_l is convolutional layer to make x_l of the same dimension as x_{l+1}.

Cross distillation provides a more complete target for the hole regions, but shallower layers might struggle to achieve it initially. Self distillation provides a more task-related and achievable target for those layers. As the training progresses, the deeper layers will be benefited more from cross-distillation, which will, in turn, improve the operation of self-distillation. The information will gradually flow from deeper to shallower layers. As shown in Network Analysis section, both of these losses, when used together, result in the best performance. The final distillation loss can be expressed as

$$R_{distil} = \sum_{l=1}^{L} R^l_{cross} + \sum_{l=1}^{L-1} R^l_{self} \quad (4)$$

Adaptive Completion-block (CB) in Encoder: CD and SD provide a target R_{distil} to the IN encoder. But, how easy is it to mimic the 'ideal' behavior? The encoder layers are expected to extract meaningful features from the uncorrupted regions as well as fill the holes with coarse information accurately, which are two different tasks. Instead of relying on the same encoder to do both, we disentangle these two tasks and use a dedicated block for updating the holes. Mask-based convolution is proposed in [18, 41], but they still utilize the same layers for both regions. [42] uses separate normalization techniques for the two regions. But extending such an idea to our approach is not straightforward. Although intuitive, simply using parallel layers in the whole encoder will significantly increase the number of parameters and be sub-optimal due to the static nature of standard CNN layers.

The completion block (CB) is used at the end of each encoder level to specifically update the hole regions. If x'_l is the feature output of l^{th} encoder level, the output of the CB block can be expressed as

$$x_l = x'_l + f_{ada-conv}(x'_l) \odot M \quad (5)$$

where $f_{ada-conv}$ is the transformation applied in the completion block. Elementwise multiplication with M ensures that it only updates the holes.

It is common to use dilated convolutions, convolutions with large kernels, or multi-scale structure [36] in inpainting networks to extract information from a large neighborhood while filling the holes. These approaches have two major limitations: (a) Required computations increase due to multiple convolutional layers. (b) The static CNN layers fail to handle the varying need of diverse hole regions. Instead, we design our CB with adaptive convolution layers ([28], [33]), where each pixel can decide where to look for information in the neighborhood (spatially-varying offset) and how much importance to give to different regions (spatially-varying weights).

Let $y = f_{ada-conv}(x'_l)$ be the output of the adaptive convolutional layer. Given input feature map $x'_l \in \mathbb{R}^{C \times H \times W}$, we use two convolutional layers to generate a spatially varying kernel V with offsets Δ and perform the convolutional operation as

$$y_j = \sum_{k=1}^{K} (V_{j,j_k})(x'_l[j + j_k + \Delta j_k]) \quad (6)$$

where K is the kernel size, j defines the output pixel location, j_k defines position of the convolutional kernel of dilation 1. $V_{j,j_k} = f_{ker}(x'_l) \in \mathbb{R}^{K^2 \times H \times W}$ and $\Delta j_k =$

$f_{off}(x'_l) \in \mathbb{R}^{2K^2 \times H \times W}$ are the learnable pixel dependent kernel and offsets, respectively. f_{ker} and f_{ker} represents convolutional layers. The kernels (V) and offsets (Δ) vary from one pixel to another, but are fixed for all the channels promoting efficiency. We also use shared weight and offset prediction layers for all the completion blocks.

3.2. Attention Transfer using Distillation in Decoder

Attention modules are widely used in inpainting literature [40, 16, 45] for its affinity-finding property. It generally divides the input feature map into small patches/pixels, finds the feature similarity of the holes with rest of the image, and then refines the holes by taking a weighted sum of the similar features, where the weight is measured by the similarity. The inherent assumption is that the features coming to the attention module are coarsely completed. If the encoder encodes wrong information for a hole region, the attention module will further aggregate similar wrong information from the entire image, leading to erroneous results. This problem becomes significant for complex big holes and hole boundaries. Recently, [31] also reported similar behavior and deployed a parallel coarse branch to force the hole regions to be filled coarsely before feeding it to the attention module.

We address this problem by directly making the attention module mimic an ideal affinity-finding behavior from the AN. At every decoder level (of both AN and IN), we deploy an attention module. As the AN's feature maps are uncorrupted, the measured pairwise similarity between different patches/pixels will be accurate, even for the hole regions. We exploit this knowledge and make the inpainting model behave similarly.

Instead of convolution based attention module of [40] which measures patch-similarities, we calculate pixel-similarities ([35, 45]) using much efficient matrix-multiplication based operations while giving equivalent performance. Note that, our proposed scheme can be easily extended to other variants of attention module as well and we analyse some of the possibilities in Sec. 4.2. Given decoder feature d_l at level l, the pairwise weightage can be calculated as

$$r_{i,j} = \frac{\exp(p_{i,j})}{\sum_{i=1}^{N} \exp(p_{i,j})} \tag{7}$$

where $p_{i,j} = f_Q(d^i)^T f_K(d^j)$, f_Q and f_K are 1x1 convolutional layers, N is the total number of pixels, d is the input feature. The output of this attention layer is given by

$$a_i = \sum_{j=1}^{N} (r_{i,j} d_j) \tag{8}$$

If $r_{i,j}^{AN}$ is the relation learned in AN branch, we minimize the distance between the IN similarity matrix ($r_{i,j}$) and the AN similarity matrix ($r_{i,j}^{AN}$). We further allow each pixel to adaptively select the relative importance of the learned similarity from the AN network using a meta-network t, consisting of fully connected layers. The attention transfer loss can be calculated as

$$R_{att} = \sum_{i \in H} t_i(d_i^{AN}) (\sum_{j=1}^{N} (r_{i,j}^{AN} - r_{i,j})^2) \tag{9}$$

where H denotes the hole region indicated by '1' in mask M, t_i lies between 0 and 1 and is generated using a single convolutional layer followed by sigmoid operation.

3.3. Pixel-adaptive Global-Local Fusion (PGL):

We have observed that directly deploying an attention module in the decoder may result in repetitive and ambiguous content, like discontinuities among the missing and background regions. As shown in Fig. 7, it results in curved edges near boundaries or repetitive tree leaves. The need for global-local consistency was highlighted by some of the earlier works like [14], which proposed the use of two separate discriminators. While attention is good at modeling long-range global context, it is less capable of extracting fine-grained local feature patterns. On the other hand, CNN layers learn shared position-based kernels over a local window that maintains translation equivariance and can capture features like edges and shapes more accurately. We hypothesize that both global and local interactions are important for filling the missing regions and thus allow each pixel to adaptively select the relative weightage for a branch. The fused output can be expressed as

$$O_{GL} = f_{att}(D) \odot W + f_{ada-conv}(D) \odot (1 - W) \tag{10}$$

where $W = \text{sigmoid}(f_w(D))$ is the per-pixel weightage map, f_w is a single convolutional layer, D is the input feature map, O_{GL} is the fused output, f_{att} and $f_{ada-conv}$ are the global attention and local adaptive convolution modules respectively. Even though attention has already been used extensively, the combination of attention and adaptive convolutions is significantly better than attention alone and produces much better locally consistent results.

Training the Meta Networks: We use several meta networks to allow the IN to take only useful guidance and reject others. These meta networks are applied to the output of the AN (Eqs. 2,3,9) and only used for training. There are four sets of meta networks: transformation γ finetunes the AN network features to be more helpful for inpainting while ρ_c, ϕ_c and t give more weightage to the useful features. We update the meta-networks intending to maximize the inpainting performance. We follow the standard approach of bilevel scheme [4, 6, 7, 8] to train the meta-networks parameterized by Ψ. The training scheme is formally outlined in Algorithm 1. We have used $T = 2$ in our experiments. Specifically, we use Reverse-HG [7] and alternatively update the inpainting model parameters θ and the meta-network parameters Ψ.

Algorithm 1: The overall training procedure.

Dataset $D_{train} = (x_i, y_i)$, learning rate α;
Total Loss $\mathcal{L}_{tot} = \mathcal{L}_{inp} + R_{distil} + R_{att}$;
while *Not Done* **do**
  Sample a batch $\mathcal{B} \subset D_{train}$ with $|\mathcal{B}| = B$;
  for $i \leftarrow 0$ **to** $T - 1$ **do**
    Update θ to minimize
    $\frac{1}{B} \sum_{(x,y)\in\mathcal{B}} \mathcal{L}_{tot}(\theta|x, y, \phi)$;
  end
  Update Ψ using $\nabla_\Psi \frac{1}{B} \sum_{(x,y)\in\mathcal{B}} \mathcal{L}_{inp}(\theta_T|x, y)$
end

4. Experiments

Experiment Setup: We evaluate our methods on Places2 [46], CelebA [22] and Paris StreetView [5] datasets. We use irregular masks from [18]. The irregular mask dataset contains 12000 irregular masks and the masked area in each mask occupies 0-60% of the total image size. We train our model with batch size 6 using the Adam optimizer and learning rate of $1e^{-4}$. The approximate number of iterations for CelebA and PSV are 400,000 where as for Places2 it is 2,500,000. All experiments are conducted using Pytorch on an Ubuntu 16 system, i7 3.40GHz CPU and an NVIDIA RTX2080Ti GPU.

Loss: The AN network is pre-trained with simple $L1$ loss. For the inpainting network, the final loss can be expressed as

$$R_{inp} = \lambda_r R_{recon} + \lambda_d (R_{distil} + R_{att}) + \lambda_s R_{style} \quad (11)$$

where R_{recon} represents the spatially-varying reconstruction loss [36] and R_{style} represents standard style loss [3]. For hyper-parameters, we found 1 for λ_r, 20 for λ_d and 150 for λ_s to give optimal results. We also use same discriminators as [42].

Comparison Models: We compare our approach with several state-of-the-art methods. These models are trained until convergence with the same experiment settings as ours. These models are: PIC [45], PC [18], GC [41], EC [24], SF [29], RFR [16] and EQ [19].

Quantitative Comparisons: We test all models on the official validation split of Places2, CelebA, and Paris StreetView datasets. We compare our model quantitatively in terms of peak signal-to-noise ratio (PSNR), structural similarity index (SSIM), and mean l_1 loss. Table 1 lists the results with different ratios of irregular masks for the three datasets. The missing results in the table are due to the limitation in computational resources. As shown in Table 1, our method produces excellent results and comfortably surpasses all the comparing models.

Qualitative Comparisons: Figs. 4, 5, 6 compare our method with previous state-of-the-art approaches on the Places2, Paris StreetView, and CelebA datasets, respectively. Our inpainting results have significantly fewer noticeable inconsistencies in most cases, especially for large holes. Compared to the other methods, our model outperforms the-state-of-the-art with more consistent colors and structures. Additionally, our model works well in real use cases as shown in Fig. 8.

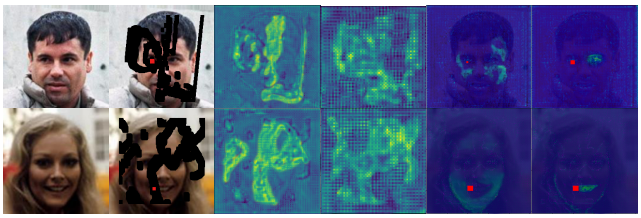

Figure 3: Visualization of adaptive filter and attention behaviors.

4.1. Network Analysis

We perform the following experiments, as reported in Table 3 on Places2 dataset. Baseline: Backbone trained with R_{recon} and R_{style} only. Net1: Baseline + CD. Net2: Baseline + SD. Net3: Baseline + SD + CD. Net4: Net3 + completion blocks. Net5: Baseline + CB. Net6: Net4 + general attention module. Net7: Net6 + attention-transfer. Net8: Net7 + PGL. Note that, for a fair comparison with our final model Net8, we have added the same number of parameters in the baseline to compensate for the few layers of completion blocks and adaptive filters in decoder.

Distillation in encoder: For Net1 and Net2, the addition of cross and self distillation losses, respectively, shows significant improvement over baseline. Using both SD and CD (Net3) performs better than the individual techniques, reinforcing the need for self and cross distillation in an entangled manner. Net4 with adaptive-convolution based completion block shows improvement over Net3. Interestingly, Net5, which uses only completion blocks without supervision, shows sub-optimal improvement, showing that distillation and completion blocks complement each other. As shown in Fig. 1, we observe that inpainting encoder's features With distillation (Column 4) are more complete and visually show correlation with the uncorrupted AN features.

Distillation in attention module: Net6 deploys attention modules in the decoder and the improvement observed shows the utility of such a module for refining coarse features. In Net7, we use additional attention transfer technique. Along with the quantitative gains of Net7 over Net6, we visualize the attention map for a particular degraded pixel in columns 4 and 5 of Fig. 3. We observe that holes near the boundary between two different regions are most prone to error while using a standard attention module and

Dataset		Places2				CelebA				Paris Street View			
Mask Ratio		10-20%	20-30%	30-40%	40-50%	10-20%	20-30%	30-40%	40-50%	10-20%	20-30%	30-40%	40-50%
PSNR↑	PIC	27.14	25.01	21.72	19.04	30.71	27.75	24.68	20.37	29.28	27.11	23.85	22.97
	PC	27.28	25.22	22.12	20.10	32.73	29.88	26.71	23.36	30.81	28.93	25.39	23.14
	GC	27.15	25.18	22.04	20.12	32.59	29.84	26.68	23.41	31.32	29.14	25.55	23.08
	EC	27.19	25.21	22.18	20.23	32.48	29.79	26.66	23.44	31.22	29.19	25.57	23.19
	SF	28.34	26.29	23.31	21.30	33.25	30.44	27.24	23.95	31.88	29.78	25.78	23.30
	RFR	-	-	-	-	33.52	30.68	27.70	24.38	31.95	29.85	26.12	23.61
	EQ	28.75	26.71	23.68	21.82	-	-	-	-	32.61	30.31	26.77	24.21
	Ours	**29.15**	**27.20**	**24.21**	**22.35**	**34.14**	**31.36**	**28.49**	**25.07**	**33.05**	**30.87**	**27.27**	**24.72**
SSIM↑	PIC	0.930	0.816	0.781	0.602	0.965	0.891	0.880	0.720	0.930	0.839	0.765	0.609
	PC	0.934	0.873	0.823	0.670	0.972	0.930	0.918	0.846	0.946	0.866	0.815	0.682
	GC	0.939	0.861	0.813	0.674	0.973	0.920	0.914	0.839	0.953	0.871	0.829	0.692
	EC	0.933	0.860	0.820	0.672	0.975	0.922	0.915	0.851	0.950	0.870	0.829	0.698
	SF	0.936	0.865	0.823	0.679	0.978	0.933	0.936	0.864	0.954	0.890	0.848	0.708
	RFR	-	-	-	-	0.979	0.934	0.929	0.880	0.955	0.891	0.849	0.712
	EQ	0.939	0.880	0.829	0.698	-	-	-	-	**0.959**	0.897	0.851	0.716
	Ours	**0.942**	**0.891**	**0.841**	**0.708**	**0.981**	**0.941**	**0.943**	**0.902**	**0.959**	**0.899**	**0.861**	**0.722**
Mean l_1↓	PIC	0.0161	0.0312	0.0441	0.0830	0.0111	0.0217	0.0314	0.0639	0.0140	0.0273	0.0379	0.0722
	PC	0.0154	0.0304	0.0409	0.0740	0.0083	0.0154	0.0236	0.0462	0.0123	0.0265	0.0313	0.0673
	GC	0.0150	0.0301	0.0405	0.0748	0.0088	0.0158	0.0245	0.0470	0.0120	0.0260	0.0309	0.0679
	EC	0.0157	0.0304	0.0408	0.0742	0.0088	0.0160	0.0247	0.0471	0.0110	0.0254	0.0286	0.0658
	SF	0.0163	0.0316	0.0418	0.0712	0.0089	0.0162	0.0244	0.0469	0.0119	0.0257	0.0290	0.0653
	RFR	-	-	-	-	0.0078	0.0156	0.0232	0.0458	0.0110	0.0252	0.0281	0.0650
	EQ	0.0149	0.0291	0.0381	0.0694	-	-	-	-	0.0108	0.0248	0.0277	0.0645
	Ours	**0.0131**	**0.0227**	**0.0344**	**0.0614**	**0.0069**	**0.0138**	**0.0201**	**0.0418**	**0.0101**	**0.0224**	**0.0235**	**0.0606**

Table 1: Numerical comparisons on three datasets. ↑ Higher is better. ↓ Lower is better.

Figure 4: Qualitative results on Places2.

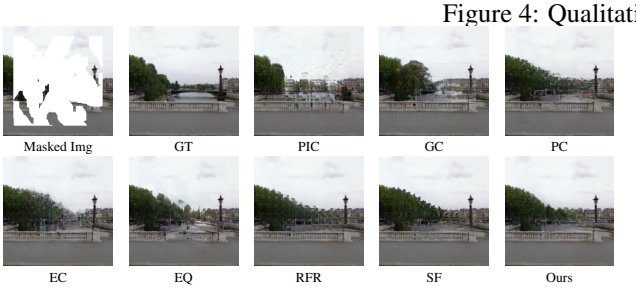

Figure 5: Qualitative comparisons on Paris Street-View dataset.

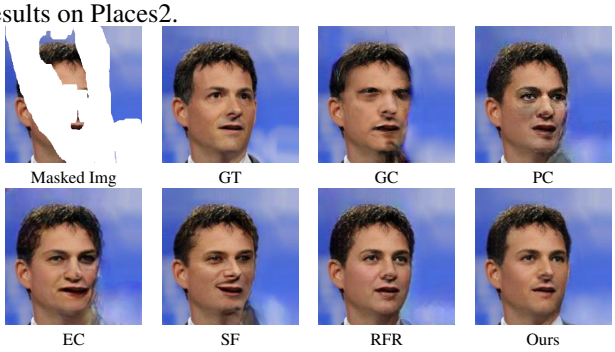

Figure 6: Qualitative comparisons on CelebA dataset.

	PC	EC	PIC	EQ*	RFR	SF*	Ours
Time (ms)	50	110	70	110	200	140	90
Params (M)	33	27	7.6	130	31	93	13

Table 2: Comparisons on runtime and model parameters.

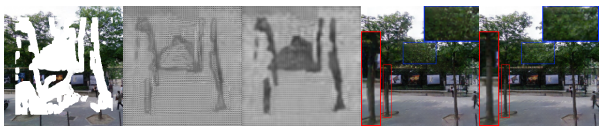

Figure 7: Comparison results for pixelwise global-local consistency (PGL).

may lead to the accumulation of wrong global information. In row 1 of Fig. 3, we visualize the attention map for a missing region in left eye, which lies very close to the eye and skin boundary. For attention without distillation (column 5), the missing region is wrongly considered as a skin pixel and the network scouts for other skin regions to fill it. For the proposed attention with distillation (column 6), it is able to find correct affinity relation and gather information from the other eye region.

Figure 8: Results on real images.

Pixel-adaptive global-local feature fusion (PGL): Net8, which is our final model, utilizes pixelwise adaptive fusion of global and local information. We visualize the pixelwise weight map W in columns 2 and 3 of Fig. 7, where lighter color represents more weightage to attention module. For initial layers of decoder (column 2), we can see that most of the hole regions prefer the attention module, which is useful for refining inner regions of big holes. But, as we go deeper, regions with high texture or sharp edges are leaning towards local consistency, which aids in processing the collected global information (from earlier stages) and merging with the neighborhood. We show the final output with and without using PGL in columns 5 and 4. Without PGL, we can observe that the highlighted tree region's texture is repetitive, and the tree on the left shows inconsistency. These regions are more accurately reproduced with the help of PGL (column 5).

To analyze the adaptive filtering module's behavior, we calculate the variation of weight and offset of the filters and plot the spatial distribution as a map (Fig. 3, column 3 and 4). Although the computed filter wights are not directly interpretable, it can be seen that the variation of the filters agrees with the degradation. We also observe that, regions near the holes show higher offset variation which denotes that those regions are indeed looking into a larger local neighborhood compared to a static CNN layer.

In Table. 2, we have compared the runtime and number of parameters of various approaches. * denotes methods that requires pre-computation of structure information, which is very slow and we have reported only the runtime of the final inpainting network. Compared to existing approaches our final model Net8 achieves much better performance with significantly less parameters.

4.2. Generalization Experiments

We perform the following experiments to verify the generalizability of the distillation based approach. We apply our distillation technique (CD, SD coupled with CB while keeping the total no. of parameters same) to some existing works: GC, EC, and SF. EC and SF are two-stage networks, where the first stage is used to predict the edges and structure. Thus, we add supervision only in the final stage EC and SF. For GC, regions of the coarse result is not obvious, so we add the supervision in the coarse network. These experiments are denoted by $*$ in Table 4. The significant performance boost shows that the distillation technique can be directly used to improve existing works without requiring extra parameters. Next, to verify the advantage of adaptive feature level supervision compared to handcrafted edges/structures, we remove the first stage from EC and SF and only use the second stage for inpainting along with distillation losses and CB. These experiments are denoted by $\#$ in Table 4. Although the number of parameters is reduced by almost 50% (by removing the first stage), utilizing the additional supervision, these methods can still achieve a performance close to the original one (which uses two stages). † indicates the experiment where we added the attention transfer technique to the contextual attention used in GC. The observed improvement proves the utility of the proposed attention transfer technique for any standard attention module.

Methods	CD	SD	CB_a	Att	AttD	PGL	PSNR
Baseline							22.36
Net1	✓						22.84
Net2		✓					22.79
Net3	✓	✓					22.98
Net4	✓	✓	✓				23.54
Net5			✓				22.54
Net6	✓	✓	✓	✓			23.62
Net7	✓	✓	✓	✓	✓		24.08
Net8	✓	✓	✓	✓	✓	✓	**24.21**

Table 3: Network Analysis on Places2 dataset (30-40% mask). CD, SD, CB_m,CB_a, Att, AttD, PGL denotes cross-distil., self-distil., CB with multi-scale conv, CB with adaptive conv, attention module, attention transfer, pixel-adaptive global-local fusion, respectively.

	GC	GC*	GC†	EC	EC*	EC#	SF	SF*	SF#
PSNR	22.04	22.74	22.41	22.18	22.62	22.06	23.31	23.70	23.19
SSIM	0.813	0.831	0.824	0.820	0.832	0.818	0.823	0.833	0.807

Table 4: The results of applying distillation based supervision to different backbone networks' encoder: GC, EC and SF. The results are based on Places2 (30-40% mask).

Supplementary details: We report layerwise network description, detailed operation of different modules, pseudo-codes and additional experimental results in the supplementary material.

5. Conclusions

This work presented a method for improving inpainting performance by using knowledge distillation. We design different distillation based guidances for different layers throughout the network. Extensive comparisons, ablation studies demonstrate the superiority of the approach in both performance and efficiency. We also demonstrate how it can also be applied to existing networks.

References

[1] Hessam Bagherinezhad, Maxwell Horton, Mohammad Rastegari, and Ali Farhadi. Label refinery: Improving imagenet classification through label progression. *arXiv preprint arXiv:1805.02641*, 2018.

[2] Arnav V Bhavsar and Ambasamudram N Rajagopalan. Range map superresolution-inpainting, and reconstruction from sparse data. *Computer Vision and Image Understanding*, 116(4):572–591, 2012.

[3] Ya-Liang Chang, Zhe Yu Liu, Kuan-Ying Lee, and Winston Hsu. Learnable gated temporal shift module for deep video inpainting. *arXiv preprint arXiv:1907.01131*, 2019.

[4] Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of operations research*, 153(1):235–256, 2007.

[5] Carl Doersch, Saurabh Singh, Abhinav Gupta, Josef Sivic, and Alexei Efros. What makes paris look like paris? 2012.

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[7] Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. Forward and reverse gradient-based hyperparameter optimization. *arXiv preprint arXiv:1703.01785*, 2017.

[8] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimilano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. *arXiv preprint arXiv:1806.04910*, 2018.

[9] Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. *arXiv preprint arXiv:1805.04770*, 2018.

[10] Mengya Gao, Yujun Shen, Quanquan Li, Liang Wan, and Xiaoou Tang. Feature matters: A stage-by-stage approach for task independent knowledge transfer. 2018.

[11] Zongyu Guo, Zhibo Chen, Tao Yu, Jiale Chen, and Sen Liu. Progressive image inpainting with full-resolution residual network. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2496–2504, 2019.

[12] Sangchul Hahn and Heeyoul Choi. Self-knowledge distillation in natural language processing. *arXiv preprint arXiv:1908.01851*, 2019.

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[14] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (ToG)*, 36(4):1–14, 2017.

[15] Yunhun Jang, Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. Learning what and where to transfer. *arXiv preprint arXiv:1905.05901*, 2019.

[16] Jingyuan Li, Ning Wang, Lefei Zhang, Bo Du, and Dacheng Tao. Recurrent feature reasoning for image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7760–7768, 2020.

[17] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. Generative face completion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3911–3919, 2017.

[18] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 85–100, 2018.

[19] Hongyu Liu, Bin Jiang, Yibing Song, Wei Huang, and Chao Yang. Rethinking image inpainting via a mutual encoder-decoder with feature equalizations. *arXiv preprint arXiv:2007.06929*, 2020.

[20] Hongyu Liu, Bin Jiang, Yi Xiao, and Chao Yang. Coherent semantic attention for image inpainting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4170–4179, 2019.

[21] Yifan Liu, Ke Chen, Chris Liu, Zengchang Qin, Zhenbo Luo, and Jingdong Wang. Structured knowledge distillation for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2604–2613, 2019.

[22] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[23] Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. *arXiv preprint arXiv:1902.03393*, 2019.

[24] Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Z Qureshi, and Mehran Ebrahimi. Edgeconnect: Generative image inpainting with adversarial edge learning. *arXiv preprint arXiv:1901.00212*, 2019.

[25] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[26] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[27] Yael Pritch, Eitam Kav-Venaki, and Shmuel Peleg. Shift-map image editing. In *2009 IEEE 12th International Conference on Computer Vision*, pages 151–158. IEEE, 2009.

[28] Kuldeep Purohit and AN Rajagopalan. Region-adaptive dense network for efficient motion deblurring. In *AAAI*, pages 11882–11889, 2020.

[29] Yurui Ren, Xiaoming Yu, Ruonan Zhang, Thomas H Li, Shan Liu, and Ge Li. Structureflow: Image inpainting via structure-aware appearance flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 181–190, 2019.

[30] Stefan Roth and Michael J Black. Fields of experts: A framework for learning image priors. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 860–867. IEEE, 2005.

[31] Min-cheol Sagong, Yong-goo Shin, Seung-wook Kim, Seung Park, and Sung-jea Ko. Pepsi: Fast image inpainting

with parallel decoding network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11360–11368, 2019.

[32] Yong-Goo Shin, Min-Cheol Sagong, Yoon-Jae Yeo, Seung-Wook Kim, and Sung-Jea Ko. Pepsi++: Fast and lightweight network for image inpainting. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[33] Hang Su, Varun Jampani, Deqing Sun, Orazio Gallo, Erik Learned-Miller, and Jan Kautz. Pixel-adaptive convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11166–11175, 2019.

[34] Gregor Urban, Krzysztof J Geras, Samira Ebrahimi Kahou, Ozlem Aslan, Shengjie Wang, Rich Caruana, Abdelrahman Mohamed, Matthai Philipose, and Matt Richardson. Do deep convolutional nets really need to be deep and convolutional? *arXiv preprint arXiv:1603.05691*, 2016.

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[36] Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Image inpainting via generative multi-column convolutional neural networks. In *Advances in neural information processing systems*, pages 331–340, 2018.

[37] Wei Xiong, Jiahui Yu, Zhe Lin, Jimei Yang, Xin Lu, Connelly Barnes, and Jiebo Luo. Foreground-aware image inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5840–5848, 2019.

[38] Zhaoyi Yan, Xiaoming Li, Mu Li, Wangmeng Zuo, and Shiguang Shan. Shift-net: Image inpainting via deep feature rearrangement. In *Proceedings of the European conference on computer vision (ECCV)*, pages 1–17, 2018.

[39] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[40] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.

[41] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4471–4480, 2019.

[42] Tao Yu, Zongyu Guo, Xin Jin, Shilin Wu, Zhibo Chen, Weiping Li, Zhizheng Zhang, and Sen Liu. Region normalization for image inpainting. In *AAAI*, pages 12733–12740, 2020.

[43] Sukmin Yun, Jongjin Park, Kimin Lee, and Jinwoo Shin. Regularizing class-wise predictions via self-knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13876–13885, 2020.

[44] Haoran Zhang, Zhenzhen Hu, Changzhi Luo, Wangmeng Zuo, and Meng Wang. Semantic image inpainting with progressive generative networks. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 1939–1947, 2018.

[45] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Pluralistic image completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1438–1447, 2019.

[46] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.